

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the Application of:

HYE KYEONG PARK, ET AL.

Application No.:

Filed:

For: **METHOD FOR PROVIDING QoS
(QUALITY OF SERVICE) -
GUARANTEEING MULTI-PATH AND
METHOD FOR PROVIDING DISJOINT
PATH USING THE SAME**

Art Group:

Examiner:

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

REQUEST FOR PRIORITY

Sir:

Applicant respectfully requests a convention priority for the above-captioned application, namely:

COUNTRY	APPLICATION NUMBER	DATE OF FILING
Republic of Korea	2002-51097	28 August 2002

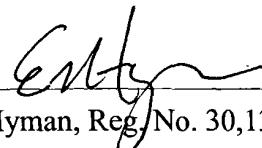
☐ A certified copy of the document is being submitted herewith.

Respectfully submitted,

Blakely, Sokoloff, Taylor & Zafman LLP

Dated: 7/18/03

12400 Wilshire Blvd., 7th Floor
Los Angeles, California 90025
Telephone: (310) 207-3800


Eric S. Hyman, Reg. No. 30,139

KOREAN PATENT ABSTRACTS(KR)

Document Code:A

(11) Publication No.1020010003353

(43) Publication Date. 20010115

(21) Application No.1019990023623

(22) Application Date. 19990623

(51) IPC Code:

H04L 12/28

(71) Applicant:

AN, SUN SHIN

KOREA TELECOM

YOON, YOUNG HYUN

(72) Inventor:

AN, SUN SHIN

CHOI, EUN HO

JUN, HONG BEOM

KIM, DU SEOK

YOON, YOUNG HYUN

(30) Priority:

(54) Title of Invention

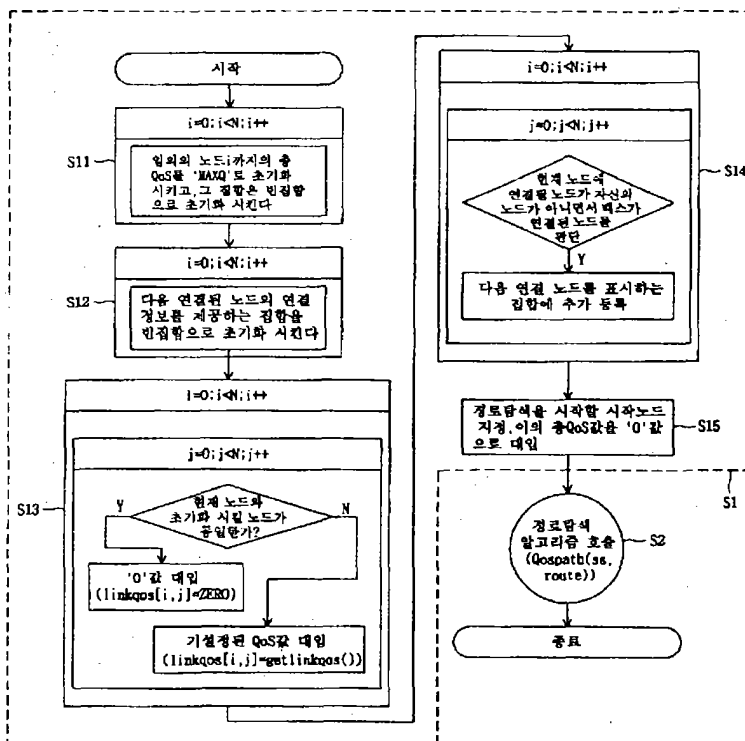
METHOD FOR DETECTING PATH SUPPORTING QUALITY OF SERVICE

Representative drawing

(57) Abstract:

PURPOSE: A method for detecting a path supporting quality of service(QoS) is provided to decide whether path information passing between a source node and a destination node is retransmitted by each node not to transmit the path information to a transmitted node, to transmit the path information to an adjacent node, and to transmit an only path request mostly suitable for a requested QoS to the adjacent node to delete or store the path information, so as to prevent a CPU and a bandwidth from being wasted.

CONSTITUTION: A GSPA(Generic Shortest Path Algorithm) supplies the shortest path detection applied to a single layer, and detects a path having at least a quality of service(QoS) in every path which can be generated as one





destination. A DGSPA(Distributed Generic Shortest Path Algorithm) distributes the path to rapidly detect the path having at least a quality of service(QoS). A QRDGSPA(QoS Restricted Generic Shortest Path Algorithm) sets up a QoS value to the shortest path when the QoS value generated from the every path satisfies a reference QoS value, and prevents set-up one path from being in an error state. Each algorithm(HDGSPA,QRHDGSPA) for applying the DGSPA and the QRDGSPA to a multi-level supplies the shortest path detection applied to a multi-level layer.

COPYRIGHT 2001 KIPO

if display of image is failed. press (F5)



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.
H04L 12/28

(11) 공개번호
(43) 공개일자

특2001-0003353
2001년01월15일

(21) 출원번호	10-1999-0023623
(22) 출원일자	1999년06월23일
(71) 출원인	한국전기통신공사, 이계철 대한민국 463-010 경기도 성남시 분당구 정자동 206 안순신 대한민국 158-050 서울특별시 양천구 목동아파트 1011동 201호 윤영현 대한민국 121-022 서울특별시 마포구 공덕2동 188-108 현대아파트 1동 705호
(72) 발명자	안순신 대한민국 158-050 서울특별시양천구목동아파트1011동201호 윤영현 대한민국 121-022 서울특별시마포구공덕2동188-108현대아파트1동705호 최은호 대한민국 305-390 대전광역시유성구전민동463-1 전홍범 대한민국 305-390 대전광역시유성구전민동463-1 김두석 대한민국 305-390 대전광역시유성구전민동463-1
(74) 대리인	이정훈
(77) 심사청구	있음
(54) 출원명	서비스 품질(Q o S)을 지원하는 경로 탐색 방법

요약

본 발명은 각종 정보통신 네트워크 상에서 멀티미디어 정보 전송시 요구되는 서비스 품질(QoS) 기반의 라우팅(Routing)을 위한 경로 탐색 방법에 관한 것인 바, 단일 계층에 적용되는 최단 경로 탐색을 제공하는 알고리즘으로, 하나의 목적지로 발생할 수 있는 모든 경로 중 QoS가 가장 적은 경로를 탐색하는 방법(GSPA)과, 상기와 같은 경로를 탐색할 경우 이를 분산하여 빠른 시간안에 경로를 탐색할 수 있도록 하는 방법(DGSPA)과, 하나의 목적지를 두고 발생할 수 있는 모든 경로에서 발생된 QoS의 값이 기준 QoS값을 만족하면 최단경로로 모두 설정하여, 하나의 경로만을 설정하였다가 이 경로가 어려워지는 것을 방지하는 방법(QRDGSPA)들을 제공하고, 다중레벨 계층에 적용되는 최단 경로 탐색을 제공하는 알고리즘으로, 상기 DGSPA알고리즘과 QRDGSPA알고리즘을 다중레벨에 적용할 수 있도록 응용한 각각의 알고리즘(HDGSPA,QRHDGSPA)을 제공함으로써, 상기 각 경로 탐색 방법 중에서 사용자가 자신의 시스템 환경에 가장 알맞는 방법을 선택적으로 사용할 수 있도록 하는 잇점이 있다.

대표도

도1

명세서

도면의 간단한 설명

도 1은 본 발명에 의한 최단 경로 탐색(GSPA) 과정을 나타내는 순서도.

도 3은 본 발명에 의한 제 1 실시예로, 분산환경을 지원하는 최단경로 탐색(DGSPA)과정을 나타내는 순서도.

도 4는 도 3의 경로 탐색 과정의 상세 과정을 나타내는 순서도.

도 5는 본 발명에 의한 제 2 실시예로, 제한된 QoS값을 이용한 최단 경로 탐색(QRDGSPA) 과정을 나타내는 순서도.

도 6은 도 5의 경로 탐색 과정의 상세 과정을 나타내는 순서도.

도 7은 본 발명에 의한 제 3 실시예로, 2레벨 네트워크에서의 최단경로탐색(HDGSPA) 과정을 나타내는 순서도.

도 8은 도 7의 경로 탐색 과정의 상세 과정을 나타내는 순서도.

도 9는 본 발명에 의한 제 4 실시예로, 2레벨 네트워크에서의 제한된 QoS값을 이용한 최단 경로 탐색(QRHDGSPA) 과정을 나타내는 순서도.

도 10은 도 9의 경로 탐색 과정의 상세 과정을 나타내는 순서도.

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 각종 정보통신 네트워크 상에서 멀티미디어 정보 전송시 요구되는 서비스 품질(Quality of Service : 이하 QoS라 칭한다) 기반의 라우팅(Routing)을 위한 경로 탐색 방법에 관한 것이다.

일반적인 경로 탐색 방법을 알아보면, 첫째, 거리 벡터(Distance Vector) 알고리즘을 들 수 있는 바, 이 거리 벡터(Distance Vector) 알고리즘은 인터넷에서 기존에 많이 사용되는 경로 탐색 알고리즘이다.

상기 방식을 사용한 대표적인 것으로 경로 탐색 정보 프로토콜(Routing Information Protocol : RIP)이 있다.

거리 벡터(Distance Vector) 알고리즘의 간단한 동작을 설명하면, 먼저 각 망에 연결되어 있는 각 노드에서 각각의 경로 탐색 테이블에 자신을 가리키는 거리(Distance) 값을 0로 하고, 망에 있는 다른 노드들의 거리 값은 무한대로 초기화 시킨다.

그런다음 경로를 찾기 위한 각 단계에서 현재 노드에서 다른 노드까지의 최단 거리를 찾아서 라우터 테이블 N에 등록한다. 이 때 최단 거리를 찾기 위한 공식은 다음과 같다.

$$D(v) \leftarrow \min[D(v), D(w) + L(w, v)]$$

여기서 $D(v)$ 는 현재 노드에서 목적 노드까지의 거리를 의미하며, $L(w, v)$ 는 노드 w 에서 노드 v 까지의 거리를 표시하는 것이다.

위에서 찾아진 최단 거리 정보는 자신과 연결되어 있는 인접 노드에 전달되어 지고, 이 절차는 각 노드에서 각 노드에 연결되어 있는 모든 노드에 대한 정보가 수집될 때 까지 지속된다.

그런 다음 각 노드에서 더 이상의 경로 변경 정보가 발생하지 않고, 각 연결된 링크(Link)에도 전송중인 정보가 존재하지 않을 때, 이 때를 모든 경로 정보가 수집된 것으로 인식하여 경로 정보 수집 과정을 종료한다.

이상과 같은 거리벡터(Distance Vector) 알고리즘의 가장 큰 장점은 단순하다는 점이다. 위의 수행 절차에서 보인 바와 같이 알고리즘이 매우 단순하여 구현하기가 용이하다.

반면, 거리 벡터 알고리즘의 단점은 각 노드에서의 모든 경로 정보를 수집하기 위해서 상대적으로 많은 시간과 망 대역폭(Network Bandwidth)을 소모한다는 점이다. 또한, 더 큰 단점은 네트워크 상태가 수시로 변화하는 환경에서 각 노드에서 변경된 정보가 인접 노드로 전달되는 과정에서 전송된 정보가 다른 경로를 통하여 다시 되돌아 오는 핑-퐁(Ping-Pong) 문제가 발생할 수 있다는 점이며, 각 중간 노드에서 최적의 QoS를 선정할 수 있는 기능이 제공되지 않는다는 점이다.

둘째, 링크-스테이트(Link-State) 알고리즘을 들 수 있는 바, 링크-스테이트(Link-State) 프로토콜은 "Shortest Path First"라고도 불리는 경로 탐색 알고리즘으로써, 분산 데이터베이스 모델을 포함하고 있으며 최단 경로를 찾기 위하여 "Dijkstra's Shortest Path" 알고리즘을 사용하고 있다.

이러한 링크-스테이트(Link-State) 알고리즘의 간단한 동작 과정을 설명하면, 각 로컬 노드에서 자신의 현재 링크 상태를 다른 노드들에게 제공한다(LSA : Link-State Advertisement).

각 노드의 상태 정보에는 각 노드의 동작중인 인터페이스, 해당 인터페이스를 통하여 정보가 전송될 때 전송 시간, 그리고 인터페이스가 어디로 연결되어 있는지에 대한 정보들이 포함된다.

각 노드의 상태 정보들은 다른 노드들에 플로딩(flooding) 함수를 이용하여 전송되며, 전송된 정보들은 각 노드의 링크-스테이트 데이터베이스에 저장된다.

각 노드에서는 저장된 정보를 이용하여 "Dijkstra's Shortest Path" 알고리즘을 사용하여 각 노드에서 다른 노드와의 최단 거리 경로를 찾아서 경로 탐색 테이블에 저장한다.

링크-스테이트 알고리즘은 전송제어프로토콜/인터넷프로토콜(TCP/IP)을 위한 "Open Shortest Path First(OSPF)"와 비동기전송모드(ATM)망을 위한 개인대개인 인터페이스(Private Network-to-Network Interface : PNNI) 등에 사용되고 있다.

이상과 같은 링크-스테이트 알고리즘의 단점은 거리 벡터(Distance-Vector) 알고리즘에 비하여 다소 구현하기가 복잡하고, 주기적으로 링크 상태 정보(LSA)를 제공함으로써 많은 대역폭을 낭비하게 되는 점이다.

또한, 링크-스테이트 알고리즘은 모든 경로 탐색 정보를 모아서 초기에 적합한 최적 경로를 유지하고 있다. 그러므로, 서로 다른 노드에서 다양한 멀티미디어 서비스가 요구시에는 서비스별로 다양한 QoS를 만족시켜야 하는데, 이미 모든 노드간의 경로가 설정되어 있는 링크-스테이트나 거리벡터로써는 적당하지 않는 점이 있다.

셋째, 동적 소스 경로 탐색(Dynamic Source Routing : DSR)알고리즘은 인터넷 엔지니어링 태스크 포스(IETF)의 "MANET" 그룹에서 제안한 알고리즘으로, 유무선 환경에서의 일반적인 경로 탐색 알고리즘이 아니고 기지국이 존재하지 않는 특수 무선 이동 네트워크인 "ad-hoc" 네트워크를 위한 경로 탐색 알고리즘이다. 또한, 여기서는 멀티미디어 서비스를 위한 QoS 지원이 전혀 고려되어 있지 않다.

이러한 동적 소스 경로 탐색 알고리즘의 동작 설명을 하면, 데이터 전송을 원하는 원시 노드(Source Node)에서 경로탐색 설정 요청을 목적 노드(Destination Node)로 향하여 인접한 모든 노드에 전송한다.

경로 설정 요청이 전달되는 각 중간 노드에서는 같은 경로 요청에 대하여 가장 빨리 도착한 경로 설정 요청만 전송하고, 그 후에 도착한 경로 정보는 재전송하지 않고 삭제한다.

목적지에 도착한 경로 설정 요청은 다시 원시 노드쪽으로 그동안 통과한 노드들의 리스트를 기록하여, 원시 노드에 전달되므로써, 경로 설정이 이루어 진다.

경로 설정이 이루어 지면, 전송되는 모든 데이터 패킷에는 패킷이 통과해야 하는 중간 노드 리스트 정보가 포함되어 있어, 이 순서에 따라 패킷이 전송된다.

상기와 같은 동적 소스 알고리즘은 "Ad-hoc" 무선 네트워크라는 특수 네트워크 환경에서만 동작하는 단점이 있고, 무조건 시간 지연이 짧은 경로 설정 패킷을 전송하기 때문에 멀티미디어 서비스에 대한 고려가 전혀없다는 단점이 있다.

발명이 이루고자 하는 기술적 과제

본 발명은 상기에 기술한 바와 같은 종래 문제점을 감안하여, 일반적인 유무선 통신망에서 멀티미디어 서비스를 위한 QoS를 지원하는 경로 탐색 알고리즘을 구현하는 것을 목적으로 하는 바, 원시 노드(Source Node)에서 목적 노드(Destination Node)사이 중간에 통과한 경로 정보를 포함하여, 한 번 지나간 노드에 경로 정보가 재전송되지 않도록 각 노드에서 재전송 여부를 결정하여 인접 노드에 경로 정보를 전송하도록 하고, 각 중간 노드에서는 같은 경로 요청에 대하여 요구되는 QoS에 제일 적합한 경로 요청만을 인접 노드에 전송하며, 그 이후에 도착한 경로 정보는 재전송하지 않고 삭제하거나 필요에 따라 저장함으로써, 재전송으로 인한 중앙처리장치(CPU) 및 대역폭(Bandwidth) 낭비를 방지하는 것을 목적으로 한다.

또한, 목적지에 도착한 각 경로 정보들은 1개 이상이 될 수 있으며, 이 때 목적 노드에서는 원하는 경로를 선택하여 다시 원시 노드를 향하여 리플라이(Reply) 메시지를 송신하면 이 리플라이 메시지는 메시지에 등록되어 있는 중간 노드를 경유하여 원시 노드에 도착하므로써, 경로 설정이 이루어 지도록 하며, 네트워크상의 모든 노드가 동시에 모든 경로를 찾기 위하여 경로 설정 요청 메시지를 전송하면, 각 노드에서는 리플라이 메시지를 전송하지 않고 각 목적 노드에서 전송된 경로 설정 요청 메시지를 리플라이로 인식하여 경로를 설정함으로써 경로 설정 시간을 단축할 수 있도록 하는 것을 목적으로 한다.

또한, 목적 노드에 도착한 여러 경로 정보는 목적에 따라 백-업(Back-up) 경로로 사용하거나 멀티 경로를 이용한 전송 속도 증가를 위해 사용할 수 있으며, 이는 경로 설정 요청자의 의도에 따라 선택할 수 있도록 하는 경로 탐색 방법을 제공하는 것을 목적으로 한다.

발명의 구성 및 작용

상기와 같이 동작되도록 하는 본 발명에 의한 정보통신 네트워크 상에서의 경로 탐색 방법은, 단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 가장 최단 경로를 탐색하기 위해 상기 네트워크 상에 존재하는 모든 경로에 있는 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색 할 시작점을 지정하여 실제 경로 탐색을 수행하도록 하는 제 2 과정을 구비하는 것을 특징으로 한다.

또한, 상기와 같이 동작되도록 하는 본 발명에 의한 정보통신 네트워크 상에서의 경로 탐색 방법은, 단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 가장 최단 경로를 탐색하기 위해 상기 네트워크 상에 존재하는 모든 경로에 있는 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색을 수행한 후, 결과 메시지를 상기 호출 대상으로 전송하는 제 3과정을 구비하여, 경로 탐색 시간을 단축하는 것을 특징으로 한다.

또한, 상기와 같이 동작되도록 하는 본 발명에 의한 정보통신 네트워크 상에서의 경로 탐색 방법은, 단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 기존 QoS 값을 만족하는 모든 경로를 탐색하기 위해 상기 네트워크 상에 존재하는 모든 경로에 있는 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점과, 기존 QoS 값을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색을 수행한 후, 결과 메시지를 상기 호출 대상으로 전송하는 제 3 과정을 구비하는 것을 특징으로 한다.

그 상태에서 연결 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 가장 최단 경로를 탐색하기 위해 상기 다중레벨 네트워크 상에 존재하는 모든 경로에 있는 계층내 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색 수행 시 한번 경유한 계층의 영역은 재 경유하지 않으며, 경로탐색이 수행되면 결과 메시지를 상기 호출 대상으로 전송하는 제 3과정을 구비하는 것을 특징으로 한다.

또한, 상기와 같이 동작되도록 하는 본 발명에 의한 정보통신 네트워크 상에서의 경로 탐색 방법은, 상기 네트워크의 계층이 다중레벨인 네트워크 상에서 단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 기준 QoS 값을 만족하는 모든 경로를 탐색하기 위해 상기 다중레벨 네트워크 상에 존재하는 모든 경로에 있는 계층내 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점과, 기준 QoS 값을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색을 수행한 후, 결과 메시지를 상기 호출 대상으로 전송하는 제 3 과정을 구비하는 것을 특징으로 한다.

상술한 목적, 특징 및 장점은 첨부된 도면과 관련한 다음의 상세한 설명을 통하여 보다 분명해 질 것이다. 이하 첨부된 도면을 참조하여 본 발명의 실시예를 상세히 설명하면 다음과 같다.

먼저, 유선망에서 멀티미디어 서비스를 제공할 수 있도록 하는 최적 탐색 알고리즘을 구현하기 위해서는 다음과 같은 사항이 고려되어야 한다.

첫째, 멀티미디어 서비스를 위한 경로 탐색 알고리즘을 설계하는 데에 가장 중요한 문제점은 대부분의 경우에 한 파라미터를 위한 개별적인 최적의 경로는 적어도 하나가 존재할 수 있어도, 모든 파라미터를 위한 최적의 경로는 존재하지 않는다는 점이다. 예를 들어, 두개의 파라미터(대역폭과 지연)가 경로 탐색에서 고려되어질 때 대부분의 경우 대역폭을 위한 최적 경로와 지연을 위한 최적 경로는 다르게 된다. 결국, QoS를 지원하는 경로 탐색을 위해서는 설정된 경로가 원하는 QoS에 최적인지를 판단할 수 있는 결정적인(decisive) 프로토콜이 요구된다.

둘째, 무선망에서 동작하는 이동 단말기들은 수시로 이동이 가능하다는 점이다. 이동하는 단말기는 네트워크 형상(topology)을 수시로 변경시켜, 통신 경로를 경로 탐색 요청 이전에 설정하는 정적 경로 탐색(Static Routing) 방식은 부 적합하므로 적응적 경로 탐색(Adaptive routing) 프로토콜이 고려되어야 한다.

셋째, 무선망에서 동작하는 단말기는 유선망에서 동작하는 라우터에 비해 적은 저장 공간과 적은 컴퓨팅 능력을 보유하고 있으므로, 복잡한 계산이 요구되는 경로 탐색 프로토콜은 부 적합하며, 단순한(simple) 경로 탐색 프로토콜이 요구된다.

넷째, 무선망에서 동작하는 단말기는 분산 환경에서 동작 가능한 분산 경로 탐색 프로토콜이 운영되어야 한다. 무선망 환경에서 중앙 제어 노드(Central Node)를 이용하여 경로 탐색 경로를 설정 해주는 프로토콜은 무선 단말이 수시로 이동할 수 있다는 특성 때문에 잦은 자료 변경이 발생한다. 또한, 변경된 자료를 관리하기 위한 시간 추가별 단말기 경로 정보가 요구되어 네트워크 대역을 많이 소모시키므로 무선망 환경에서는 부 적합하다. 결국, 무선 이동통신을 위해서는 라우터가 경로 탐색을 위한 최소한의 정보를 스스로 유지 관리하는 분산 경로 탐색 프로토콜이 요구된다.

이하 본 발명에서 구현한 최단 경로 탐색 알고리즘을 설명한다.

도 1은 본 발명에 의한 최단 경로 탐색 알고리즘(Generic Shortest Path Algorithm : 이하 GSPA라 칭한다)을 나타내는 순서도로, 네트워크 상에 존재하는 모든 경로에 있는 노드들의 연결정보를 나타내는 집합 및 QoS값을 초기화 시키는 과정(S1)과;

상기 초기화가 완료되면 라우팅 할 시작점을 지정하여 실제 경로탐색을 수행하도록 하여 최적 경로를 탐색하는 과정(S2)을 구비한다.

먼저 상기 초기화 과정(S1)을 설명하면, 반복 문장을 이용하여 노드 0에서부터 노드 N까지 소스로부터 가장 짧은 라우트를 따라 노드 i에 쌓인 총 QoS 값들을 'MAXQ' 값으로 초기화 시킨다. 이 값은 이때의 노드가 패스되지 않는 노드라는 것을 의미한다.

그리고 가정된 가장 짧은 라우트를 따라 소스로부터 노드 i에 정리된 노드의 집합 역시 빈 집합{@}으로 초기화 시킨다(S11).

참고로 이때의 알고리즘과 변수 정의는 다음과 같다.

```
struct { Qos tqos; Oset set;} node[N]; /* 노드 정보 */
Oset nextset[N]; /* 노드의 정리된 이웃 집합 */
Qos linkqos[N, N]; /* 각 노드들간의 QoS 정보 */
typedef struct { Qos tqos; Oset set; } Route;
Main(int ss) /*소스 번호*/
{
    int i, j; Route route;
    for (i = 0; i < N; i++) {node[i].tqos = MAXQ; node[i].set = {@};}
```

그런 다음 현재의 노드와 다음 연결될 노드의 연결 정보를 제공하는 집합을 초기화 시키는 바, 이 역시 반복 문장을 이용하여 빈 집합으로 초기화 시킨다(S12).

참고로 이때의 알고리즘은 다음과 같다.

```
for (i = 0; i < N; i++) nextset[i] = {@}; /* {@} : 빈 집합*/
```

이여 임의의 노드 i와 그 옆의 임의의 노드 j사이에 연결되는 링크의 QoS 값을 초기화 시키는 바, 이 역시 노드 전체에 대해 반복 문장을 이용하여 QoS값을 대입하는데 현재 초기화 시킬 노드가 자신의 노드인지를 판단하여 자신의 노드가 아니면 기 설정된 QoS값을 대입하고, 자신의 노드이면 '0'값을 대입한다(S13).

참고로 이때의 알고리즘은 다음과 같다.

```
for (i = 0; i < N; i++) for (j = 0; j < N; j++) {
    if (i == j) linkqos[i, j] = ZERO else linkqos[i, j] = getlinkqos();
}
```

이여 상기에서 대입한 연결 QoS값을 이용하여 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축하는데, 이 역시 모든 노드에 대해 반복 문장을 이용하여 연결 집합을 구축한다.

즉, 현재의 노드에서 다음에 연결될 노드가 자기 자신의 노드가 아니면서 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록한다(S14).

참고로 이때의 알고리즘은 다음과 같다.

```
for (i = 0; i < N; i++) for (j = 0; j < N; j++) {
    if (linkqos[i, j] != ZERO && linkqos[i, j] != MAXQ) nextset[i] = nextset[i] U j;
}
```

/ *연결됨: Qos값, 연결안됨: MAXQ, 이웃 집합 구축, U: 정리된 집합 */

이여 시작점을 지정한 다음 이 시작점부터 경로 탐색을 시작하여 최단 경로를 탐색하도록 하는 바, 라우트 집합을 시작점으로 초기화시키고, 라우트 QoS 값 역시 '0'값으로 시작하여 경로 탐색을 시작할 수 있도록 한다(S15).

이상 모든 초기화가 완료되면 상기 경로 탐색 시작점부터 시작하여 경로 탐색을 수행하도록 경로 탐색 알고리즘(Qospath(ss, route))을 호출한다(S2).

참고로 이때의 알고리즘은 다음과 같다.

```
route.set = {ss}; route.tqos = ZERO;
```

```
Qospath(ss, route);
```

도 2는 상기 호출에 따라 동작하는 경로 탐색 알고리즘(Qospath(ss, route))에 따라 최적 경로를 탐색하는 과정(S2)을 나타내는 순서도로, 현재 까지 경로 탐색된 노드까지의 총 QoS값을 계산한다(S21).

참고로 이때의 알고리즘 및 변수 정의는 다음과 같다.

```
Qospath(int s, Route route)
```

```
{
    Qos tqos;
    int i;
    tqos = F(route.tqos, linkqos[E(route.set), s]);
```

상기에서 경로 탐색된 현재 노드까지의 총 QoS값이 계산이 되었으면, 이때의 총 QoS값과 다른 경로를 통해 현재 노드까지 도달하여 기 계산되었던(현재까지의 노드에 도착하는 경로는 여러 경로가 존재하기 때문임) 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기한다($C(tqos) < C(node[s].tqos)$)(S22).

상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 이 경로가 지난번 경로보다 최단 거리가 되므로 경로 탐색 정보를 이 경로로 수정해야 하는 바, 먼저 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 확인한다($E(route.set) != s$)(S23).

상기 판단결과 자신의 노드가 아니면 이 노드를 연결 정보에 추가 등록시키고 다음 과정(S25)을 수행한다(S24).

그리고 상기 판단결과 자신의 노드이면 상기 과정(S24)을 생략하고 곧바로 현재까지의 경로에 대한 총 라우트 QoS값을 계산하는데, 이는 연결 정보를 추가할 경우 자신의 노드는 추가시 중복되므로 생략하는 것이다.

상기 라우는 경로를 내보낸다(S25).

참고로 이때의 알고리즘은 다음과 같다.

```
if (C(tqos) < C(node[s].tqos)) { /* C: 주어진 Qos에 대한 코스트(cost) 함수 */  
    if(E(route.set) != s) route.set = route.set U s;  
  
    route.tqos = tqos;  
  
    node[s].set = route.set;
```

참고로 상기 함수 E()는 마지막 노드와 이 노드 사이의 연결 Qos를 얻는데 사용된다

상기 과정이 완료되면 상기 호출된 노드에 대한 최적 경로 탐색이 완료되었으므로, 그 다음 노드에 대한 최적 경로 탐색을 반복 문장을 이용하여 상기과 같은 과정(S21 ~ S25)을 반복 수행한다.

이때도 역시 상기 경로 탐색 알고리즘을 호출하는 것으로 수행되는 바, 특히 이때에는 다음 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후 경로 탐색을 시작한다. 이는 한번 거쳐왔던 경로는 다시 반복하지 않도록 하기 위해서이다(S26).

참고로 이때의 알고리즘은 다음과 같다.

```
for (each i ∈ (nextset[s] route.set)) Qospath(i, route):
```

한편, 상기과 같은 방법은 경로 탐색 알고리즘을 호출할 경우 다음 노드에 대한 연결을 나타내는 집합에서 먼저 수행되었던 라우트 집합을 뺀 후 경로 탐색을 시작하는 것으로 설명하였으나, 이처럼 뺀 값이 아니라 할지라도, 본 발명의 알고리즘은 코스트 함수의 의미에서 루프 값이 양의 값을 가지면 가장 짧은 경로를 선택한다.

도 3은 본 발명에 의한 또 다른 실시예를 나타내는 최적 경로 탐색 알고리즘을 나타내는 순서도로, 이는 분산환경을 위한 알고리즘(Distributed Generic Shortest Path Algorithm : DGSPA)이다. 이 알고리즘이 상기 GSPA와 다른 점은 네트워크 상에 존재하는 모든 노드에 대해 경로 탐색을 수행하는 것이 아니라, 자신과 자신의 바로 옆에 연결된 노드에 대해서만 탐색을 수행한다.

그리고 DGSPA에서 목적 노드에 도착하는 경로 메시지 중에서 최선의 경우만을 선택하면, 하나의 경로만 탐색되지만 요구된 QoS를 만족하는 모든 경로를 선택하면 QoS를 만족하는 다중 경로를 탐색하게 된다.

이하 DGSPA의 과정을 설명한다.

임의의 하나의 노드와 상기 하나의 노드에 옆에 연결된 경로에 있는 노드들의 연결정보를 나타내는 집합 및 QoS값을 초기화 시키는 과정(T1)과;

상기 초기화가 완료된 상태에서 실제 경로 탐색을 수행하도록 하는 호출이 수신될 때까지 대기하다 호출이 수신되면 최적 경로를 탐색하는 과정(T2)을 구비한다.

먼저 상기 초기화 과정(T1)을 설명하면, 임의의 노드 s에 대한 총 QoS값을 이 노드는 패스되지 않는 노드라는 것을 의미하는 'MAXQ' 값으로 초기화 시키고, 상기 임의의 노드에 대한 정보를 가지는 값과, 이 노드와 연결될 다음 노드를 가리키는 값은 빈 집합으로 초기화시킨다(T11).

참고로 이때의 알고리즘과 변수 정의는 다음과 같다.

```
typedef struct{ Qos tqos; Oset set; } Route;  
  
struct { Qos tqos; Oset set;} node[s];  
  
Oset nextset[s];  
  
Qos linkqos[s][N];  
  
Route route;  
  
node[s].tqos = MAXQ;  
  
node[s].set = {@};  
  
nextset[s] = {@};
```

이어 상기 임의의 노드 s와 그 옆의 임의의 노드 j사이에 연결되는 링크의 QoS 값을 초기화 시키는 바, 이 역시 반복 문장을 이용하여 QoS값을 대입하는데 현재 초기화시킬 노드가 자신의 노드인지를 판단하여 자신의 노드가 아니면 기 설정된 QoS값을 대입하고, 자신의 노드이면 '0'값을 대입한다(T12).

참고로 이때의 알고리즘은 다음과 같다.

```
for (j = 0; j < N; j++) {  
    if (s == j) linkqos[s][j] = ZERO else linkqos[s][j] = getlinkqos();  
}
```

이어 상기에서 대입한 연결 QoS값을 이용하여 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축하는데, 이 역시 옆에 연결된 노드에 대해 반복 문장을 이용하여 연결 집합을 구축한다.

즉, 현재의 노드에서 다음에 연결될 노드가 자기 자신의 노드가 아니면서 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록한다(T13).

참고로 이때의 알고리즘은 다음과 같다.

```
for (j = 0; j < N; j++) {
    if (linkqos[s][j] != ZERO && linkqos[s][j] != MAXQ) nextset[s] = nextset[s] U j;
}
```

/*연결됨: Qos값, 연결안됨: MAXQ, 13), nextset[s] : s의 이웃집합*/

상기와 같이 모든 초기화가 완료되면 경로 탐색 알고리즘을 시작하게 되는데, 현재 노드가 소스노드에 해당하는지 여부를 판단하여, 소스노드이면 라우트 집합을 현재노드로 초기화하고, 라우트 QoS 값 역시 '0'로 초기화시킨 다음 경로 탐색 알고리즘을 수행하도록 호출한다(QoSpath(s, route)).

그리고 상기에서 판단결과 소스노드가 아니면 타 노드에서 경로 탐색을 완료하고 그 다음 노드의 경로 탐색을 위해 경로탐색 과정에서 결과로 전송된 값인 노드와 이때의 라우트 값이 수신(receive(s,route))되기를 대기한다. 결과값이 수신되면 경로탐색알고리즘을 호출한다(QoSpath(s, route))(T14, T2).

참고로 이때의 알고리즘은 다음과 같다.

```
if(s is the source node)
{
    route.set = {s};
    route.tqos = ZERO;
}
else
{
    receive(s, Qospath(s, route));
    call(Qospath(s, route));
}
```

도 4는 상기 호출에 따라 동작하는 경로 탐색 알고리즘(Qospath(int s, Route route))에 따라 최적 경로를 탐색하는 과정(S2)을 나타내는 순서도로, 현재까지 경로 탐색된 노드까지의 총 QoS값을 계산한다(T21).

참고로 이때의 알고리즘 및 변수 정의는 다음과 같다.

```
Qospath(int s, Route route)
{
    Qos tqos;
    int i;
    tqos = F(route.tqos, linkqos[E(route.set), s]);
```

상기에서 경로 탐색된 현재 노드까지의 총 QoS값이 계산이 되었으면, 이때의 총 QoS값과 다른 경로를 통해 현재 노드까지 도달하여 기 계산되었던(현재까지의 노드에 도착하는 경로는 여러 경로가 존재하기 때문임) 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기한다(T22).

상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 이 경로가 지난번 경로보다 최단 거리가 되므로 경로 탐색 정보를 이 경로로 수정해야 하는 바, 먼저 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 확인한다(T23).

상기 판단결과 자신의 노드가 아니면 이 노드를 연결 정보에 추가 등록시키고 다음 과정(T25)을 수행한다(T24).

그리고 상기 판단결과 자신의 노드이면 상기 과정(T24)을 생략하고 곧바로 현재까지의 경로에 대한 총 QoS값을 계산하는 바, 이는 연결 정보를 추가할 경우 자신의 노드는 추가시 중복되므로 생략하는 것이다.

즉, 경로를 나타내는 전체 QoS값에 상기에서 계산한 총 QoS값을 대입하고, 현재까지의 노드를 나타내는 집합에 상기 라우트 경로를 대입한다(T25).

참고로 이때의 알고리즘은 다음과 같다.

if(E(route.set) != s) route.set = route.set U s;

route.tqos = tqos;

node[s].set = route.set;

상기 과정이 완료되면 상기 호출된 노드에 대한 최적 경로 탐색이 완료되었으므로, 각각의 이웃한 노드에 대해 최적 경로 탐색을 위해 경로탐색(send(i, route))메시지를 보낸다. 이 메시지는 상기 초기화 과정(T1) 중 호출을 수신하는 부분(T14)으로 전송된다. 그리고 상기 메시지에 전송에 따라 호출이 수행되면 상기과 같은 과정(T21 ~ T25)을 반복 수행한다.

특히 상기 호출에 의해 경로 탐색 알고리즘이 반복 수행될 경우에는 다음 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후 경로 탐색을 시작한다. 이는 한번 거쳐왔던 경로는 다시 반복하지 않도록 하기 위해서이다(T26).

참고로 이때의 알고리즘은 다음과 같다.

```
for (each i ∈ (nextset[s] route.set))  
    send (i, route) ;
```

도 5는 본 발명에 의한 또 다른 실시예인 제한된 QoS값을 이용한 최단 경로 탐색 알고리즘(QoS Restricted Generic Shortest Path Algorithm : 이하 QRDGSPA라 칭한다)을 나타내는 순서도이다.

QRDGSPA 알고리즘은 상기 도 1에서 구현된 GSPA 알고리즘과 비교하여 볼 때 다음과 같다.

도 1에서 제시한 GSPA는 동일한 하나의 목적지에 도착하는 방법으로 최단 거리에 속하는 하나의 경로만을 선택한다.

즉, 코스트를 비교하는 과정에서 무조건 다른 경로를 통해 기 계산이 되어 있는 총 QoS값과 현재의 경로를 통해 계산된 총 QoS값을 비교하여 현재가 작으면 기존 경로를 버리고 현재의 경로를 최단 경로로 설정하므로써, 단 하나의 경로만 얻는 것이다.

그러나 상기 QRDGSPA알고리즘은 상기 코스트를 비교하는 과정에서 기준 QoS값을 설정하여 놓고 현재 경로든 다른 경로든 각각의 경로를 통해 계산된 QoS 값이 상기 설정된 기준 QoS값보다 작으면 이 경로들을 모두 최단 경로로 설정하는 것이다.

즉, 하나의 목적지가 정해지고, 이 목적지로 갈 수 있는 여러 개의 경로가 있을 경우 이 경로 모두 QoS값을 만족하면 최단 경로로 선택하는 것이다.

이렇게 되면 단 하나만의 최단 경로를 통해 경로 탐색을 하다 이 경로에 애러가 발생하게 되면 경로 탐색을 할 수 없는 문제를 방지할 수 있게 된다.

알고리즘은 도 3에 도시된 것과 거의 유사하고, 다른 부분은 경로 탐색 알고리즘을 호출 및 그 결과를 수신하는 부분에서 호출 인자가 하나 추가되는데 이는 상기에서 설명한 기준 QoS값이 된다.

즉, receive(s, route)에서 receive(s, route, req)로 바뀌고(T15), QoSpath(s, route)에서 QoSpath(s, route, req)로 바뀐 것이다(T2).

그 외의 알고리즘은 동일하므로 설명을 생략한다.

도 6은 상기 QRDGSPA에서 호출하는 경로 탐색 알고리즘을 나타내는 순서도로, 이 역시 도 4의 경로 탐색 알고리즘과 비교하여 볼 때 과정 'T1'에서 총QoS값을 계산한 후, 하기 과정이 추가된다.

즉, 소스노드에서부터 지금노드 까지의 축적된 총 QoS값이 상기 기준값으로 설정한 QoS 값을 만족하는지 여부를 검사하는 과정이 추가된다. 이 과정에서 만족하면 그 다음 경로 탐색 알고리즘을 수행하고, 만족하지 않으면 종료한다(S(tqos,req) == FALSE)(T27).

그리고 마지막에서 다시 경로 탐색 알고리즘을 반복 수행하기 위해 경로 탐색 메시지(send(i, route, req))를 전송한다(T28).

그 외의 알고리즘은 도 4와 동일하므로 설명을 생략한다.

한편, 상기까지는 네트워크의 구성이 1레벨에 해당할 경우 경로탐색 과정을 설명한 것이고 하기에서부터는 네트워크 구성이 2레벨(two level hierarchical network)일 경우 경로탐색을 수행하는 과정을 설명한다.

네트워크 구성이 2레벨일 경우 먼저 2레벨 계층 구조적 네트워크의 주소를 한 쌍의 정수인 [n, m]으로 정의한다.

여기서 n은 상위 어드레스(higher address)를 의미하고, m은 하위 어드레스(lower address)를 의미한다.

같은 상위 어드레스를 가진 노드의 집합을 영역(region)이라 하고, 특히, 계층 구조적 네트워크는 다른 상위 주소를 가진 주소 또는, 주소들로 분리된 연결 주소 리스트가 같은 상위 어드레스를 가질 수 없다.

즉, 특정 영역을 벗어난 연결은 이미 경유한 같은 영역을 다시는 경유할 수 없다는 것이다.

예를 들어 [1,2][1,4][1,3][3,1][3,2][3,3][2,1][2,2]와 같은 라우팅 어드레스 리스트는 계층 구조적 라우팅을 위한 어드레스 리스트로 적합하나 [1,2][1,4][1,3][3,1][2,1][3,2][3,3][2,2]와 같은 어드레스 리스트는 어드레스 그룹이 같은 상위 어드레스(다른 상위 어드레스)를 가진 [2,1]에 의해 격리된 [3,..]를 가지고 있기 때문에 계층 구조적 라우팅으로서는 부적합 하다.

즉, 이 연결은 노드[3,1]을 통해서 영역 3을 벗어나 노드[2,1]을 지나서 노드[3,2]를 통해 다시 같은 영역 3으로 다시 들어가는 것이므로, 한번 경유한 영역을 지나면 다시 상기 경유한 동일한 영역으로 들어갈 수 없는 점에 위배되기 때문이다.

이하 2레벨에서의 경로 탐색 방법을 설명한다.

도 7은 본 발명에 의한 제 3 실시예로, 2레벨 네트워크에서의 최단경로탐색(HDGSPA) 과정을 나타내는 순서도로, 이는 상기 설명한 DGSPA와 거의 유사하며, 다른 부분은 초기화를 시킬 경우 2레벨 계층 모두를 초기화 시켜야 한다는 점이다.

임의의 계층 i에 속하는 임의의 노드 p에 대한 총 QoS값을 이 노드는 패스되지 않는 노드라는 것을 의미하는 'MAXQ' 값으로 초기화 시키고, 상기 임의의 계층에 속하는 임의의 노드에 대한 정보를 가지는 집합과, 이 노드와 연결될 다음 계층에 속하는 노드를 가리키는 집합을 빈 집합으로 초기화시킨다(P11).

참고로 이때의 알고리즘과 변수 정의는 다음과 같다.

```
struct{ Qos tqos; Oset set;} node [H,N]; /* 계층 주소 */
```

```
Qos linkqos[[H,N],[H,N]];
```

```
Oset nextset[H,N];
```

```
struct Route { Qos tqos; Oset set; } route;
```

```
struct Node { int s; int p;};
```

```
node[i,p].tqos = MAXQ;
```

```
node[i,p].set = {@};
```

```
nextset[i,p] = {@};
```

이후 상기 임의의 계층 i에 속하는 임의의 노드 p와 그 옆의 임의의 계층 j에 속하는 임의의 노드 q사이에 연결되는 링크의 QoS 값을 초기화 시키는 바, 이 역시 반복 문장을 이용하여 QoS값을 대입하는데 현재 초기화 시킬 계층의 노드가 자신의 노드인지를 판단하여 자신의 노드가 아니면 기 설정된 QoS값을 대입하고, 자신의 노드이면 '0'값을 대입한다(P12).

참고로 이때의 알고리즘은 다음과 같다.

```
for(i = 0; i < H; i++)for(j = 0; j < H; j++){
    for(p = 0; p < N; p++)for(q = 0; q < N; q++){
        if((i,p) == (j,q))linkqos[[i,p],[j,q]] = ZERO;
        else linkqos[[i,p],[j,q]] = getlinkqos();
    }
}
```

이후 상기에서 대입한 연결 QoS값을 이용하여 자신의 노드 다음에 연결될 계층의 노드 정보를 나타내는 집합을 구축하는데, 이 역시 옆에 연결된 계층의 노드에 대해 반복 문장을 이용하여 연결 집합을 구축한다.

즉, 현재 계층의 노드에서 다음에 연결될 계층의 노드가 자기 자신의 노드가 아니면서 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록한다(P13).

참고로 이때의 알고리즘은 다음과 같다.

```
for(i = 0; i < H; i++)for(j = 0; j < H; j++){
    for(p = 0; p < N; p++)for(q = 0; q < N; q++){
        if(linkqos[[i,p],[j,q]] != ZERO && linkqos[[i,p],[j,q]] != MAXQ)
            nextset[i,p] = nextset[i,p] U (j,q);
    }
}
```

/*연결됨: Qos값, 연결안됨: MAXQ */

상기와 같이 모든 초기화가 완료되면 경로 탐색 알고리즘을 시작하게 되는데, 현재 계층의 노드가 소스노드에 해당하는지 여부를 판단하여, 소스 노드이면 경로탐색을 시작할 집합을 현재노드로 초기화하고, 그에 따른 QoS 값 역시 '0'로 초기화시킨 다음 경로 탐색 알고리즘을 수행하도록 호출한다(QoSpath(s, route)).

그리고 상기에서 판단결과 소스노드가 아니면 타 노드에서 경로 탐색을 완료하고 그 다음 노드의 경로 탐색을 위해 경로탐색 과정에서 결과로 전송된 값인 노드와 이때의 라우트 값이 수신(receive(s, route))되기를 대기한다. 결과값이 수신되면 경로탐색알고리즘을 호출한다(QoSpath(s, route))(P14, P2).

```
if(s is the source node)
```

```
{ route.set = {s};
```

```
route.tqos = ZERO;
```

```
}
```

```
else
```

```
{
```

```
receive(s, Qospath(s, route));
```

```
call(Qospath(s, route));
```

```
}
```

도 8은 상기 호출에 따라 동작하는 경로 탐색 알고리즘(Qospath(int s, Route route))에 따라 최적 경로를 탐색하는 과정(P2)을 나타내는 순서도로, 현재까지 경로 탐색된 임의의 계층 내 노드까지의 총 QoS값을 계산한다(P21).

참고로 이때의 알고리즘 및 변수 정의는 다음과 같다.

```
Qospath(Node s, Route route)
```

```
{
```

```
Qos tqos;
```

```
ini i;
```

```
tqos = F(route.tqos, linkqos[E(route.set),s]);
```

상기에서 경로 탐색된 현재 계층내 노드까지의 총 QoS값이 계산이 되었으면, 이때의 총 QoS값과 다른 경로를 통해 현재 계층의 노드까지 도달하여 기 계산되었던(현재까지의 노드에 도착하는 경로는 여러 경로가 존재하기 때문임) 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기한다(P22).

상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 이 경로가 지난번 경로보다 최단 거리가 되므로 경로 탐색 정보를 이 경로로 수정해야 하는 바, 먼저 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 확인한다(P23).

상기 판단결과 자신의 노드가 아니면 이 계층내 노드를 연결 정보에 추가 등록시키고 다음 과정(P25)을 수행한다(P24).

그리고 상기 판단결과 자신의 노드이면 상기 과정(P24)을 생략하고 곧바로 현재까지의 경로에 대한 총 QoS값을 계산하는 바, 이는 연결 정보를 추가할 경우 자신의 노드는 추가시 중복되므로 생략하는 것이다.

즉, 경로를 나타내는 전체 QoS값에 상기에서 계산한 총 QoS값을 대입하고, 현재까지의 노드를 나타내는 집합에 상기 계층내 노드 경로를 대입한다(P25).

참고로 이때의 알고리즘은 다음과 같다.

```
if (C(tqos) < C(node[s].tqos)) { /* C: 주어진 Qos에 대한 코스트(cost) 함수 */
```

```
if(E(route.set) != s) route.set = route.set U s;
```

```
route.tqos = tqos;
```

```
node[s].set = route.set;
```

상기 과정이 완료되면 상기 호출된 노드에 대한 최적 경로 탐색이 완료되었으므로, 각각의 이웃한 노드에 대해 최적 경로 탐색 메시지를 보낸다. 이 메시지는 상기 초기화 과정(P14) 중 호출을 수신하는 부분(T14)으로 전송된다. 그리고 상기 메시지 전송에 따라 호출이 수행되면 상기과 같은 과정(P21 ~ P25)을 반복 수행한다.

특히 상기 호출에 의해 경로 탐색 알고리즘이 반복 수행될 경우에는 다음 계층내 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후 경로 탐색을 시작한다. 이는 한번 거쳐왔던 계층내 경로는 다시 반복하지 않도록 하기 위해서이다.

일 예를 들면, $\{(s,p),(w,r),(f,g),(h,k)\} - \{(c,p),(w,r),(f,v),(h,r)\} = \{(s,p),(h,k)\}$ 이다. 이는 h계층을 기준으로 했을 때 s계층은 거치지 않았으므로 거쳐야 할 경로로 남고, w와 f계층은 이미 거쳤으므로 다시는 거치지 않으며, h계층내의 k노드는 아직 거치지 않았으므로 거쳐야 할 노드로 남게 되는 것이다(T26).

참고로 이때의 알고리즘은 다음과 같다.

```
for (each i ∈ {nextset[s] route.set})
```


send (i, route) ;

도 9는 본 발명에 의한 제 4 실시예로, 2레벨 네트워크에서의 제한된 QoS값을 이용한 최단 경로 탐색 알고리즘(QRHDGSPA) 과정을 나타내는 순서도로, 이는 상기 설명한 QRGSPA와 거의 유사하며, 다른 부분은 초기화를 시킬 경우 2레벨 계층 모두를 초기화 시켜야 한다는 점이다.

QRHDGSPA 알고리즘은 상기 도 5에서 구현된 QRDGSPA 알고리즘과 비교하여 볼 때 계층을 포함한 모든 노드에서 동일한 하나의 목적지에 도착하는 방법으로, 기존 QoS값을 만족하는 모든 노드들을 최적 경로로 선택하는 점이 다르다.

경로탐색 과정은 도 7에 도시된 것과 거의 유사하고, 다른 부분은 경로 탐색 알고리즘을 호출 및 결과를 수신하는 부분에서 호출 인자가 하나 추가되는데 이는 상기에서 설명한 기존 QoS값이 된다.

즉, receive(s, route)에서 receive(s, route, req)로 바뀌고(P15), QoSpath(s, route)에서 QoSpath(s, route, req)로 바뀐 것이다(P2).

그 외의 알고리즘은 도 7과 동일하므로 설명을 생략한다.

도 10은 상기 QRHDGSPA에서 호출하는 경로 탐색 알고리즘을 나타내는 순서도로, 이 역시 도 6의 경로 탐색 알고리즘(QRDGSPA)과 비교하여 볼 때 거의 유사하며, 다른 부분은 경로 탐색시 계층 탐색을 수행한다는 점과, 다음 경로 탐색을 위한 반복 탐색을 수행할 경우 한 번 경유한 영역은 다시는 거치지 않도록 하는 것이다.

탐색 과정은 도 8에 도시된 것과 거의 유사하고, 다른 부분은 과정 'P21'에서 총QoS값을 계산한 후, 하기 과정이 추가된다.

즉, 소스노드에서부터 현재 계층내 노드 까지의 축적된 총 QoS값이 상기 기준값으로 설정한 QoS 값을 만족하는지 여부를 검사하는 과정이 추가된다. 이 과정에서 만족하면 그 다음 경로 탐색 알고리즘을 수행하고, 만족하지 않으면 종료한다(S(tqos,req) == FALSE)(P27).

그리고 마지막에서 다시 경로 탐색 알고리즘을 반복 수행하기 위해 경로 탐색 알고리즘을 호출할 경우 도 8과 동일하게 경로탐색 메시지(send(i, route, req))를 전송한다(P28).

그 외의 알고리즘은 도 8과 동일하므로 설명을 생략한다.

덧붙여 상기에서 코스트 함수(C)에서 비교하는 부분 중 비교 대상이 메시지 전송에 대한 수신값을 받는 딜레이 값과, 소스노드에서 목적노드 사이에 존재하는 중간노드(hops)수가 되는 경우에는 현재 경로에서 발생한 값이 기존 경로에서 발생한 값보다 작으면 좋고, 대역폭 값 비교일 경우에는 현재 경로에서 발생한 값이 기존 경로에서 발생한 값보다 커야지만 좋다.

이러한 조건들은 각 경로탐색방법을 사용하는 사용자에게 따라 선택적으로 사용할 수 있음은 물론이며, 또한, 상기 2레벨 계층에서의 HDGSPA와 QRHDGSPA알고리즘은 설명의 편의상 네트워크가 2레벨로 되어 있을 경우를 예를 들어 설명하였으나, 이 네트워크의 레벨은 얼마든지 다중 레벨로 구성될 수 있고, 이에 따라 본 발명 역시 집합을 레벨에 따라 증가시키면 되므로 다중레벨에 적용할 수 있음은 물론이다.

발명의 효과

이상에서 상세히 설명한 바와 같이 본 발명은 각종 정보통신 네트워크 상에서 멀티미디어 정보 전송시 요구되는 QoS 기반의 경로 탐색(Routing)을 제공하기 위해, 단일 계층에 적용되는 최단 경로 탐색을 제공하는 알고리즘으로, 하나의 목적지로 발생할 수 있는 모든 경로 중 QoS가 가장 적은 경로를 탐색하는 방법(GSPA)과, 상기와 같은 경로를 탐색할 경우 이를 분산하여 빠른 시간안에 경로를 탐색할 수 있도록 하는 방법(DGSPA)과, 하나의 목적지를 두고 발생할 수 있는 모든 경로에서 발생된 QoS의 값이 기존 QoS값을 만족하면 최단경로로 모두 설정하여, 하나의 경로만을 설정하였다가 이 경로가 예러가 나는 것을 방지하는 방법(QRDGSPA)들을 제공하고, 다중레벨 계층에 적용되는 최단 경로 탐색을 제공하는 알고리즘으로, 상기 DGSPA알고리즘과 QRDGSPA알고리즘을 다중레벨에 적용할 수 있도록 응용한 각각의 알고리즘(HDGSPA,QRHDGSPA)을 제공함으로써, 상기 각 경로 탐색 알고리즘 중에서 사용자가 자신의 시스템 환경에 가장 알맞는 알고리즘을 선택적으로 사용할 수 있도록 하는 잇점이 있다.

아울러 본 발명의 바람직한 실시예들은 예시의 목적을 위해 개시된 것이며, 당업자라면 본 발명의 사상과 범위안에서 다양한 수정, 변경, 부가 등이 가능할 것이며, 이러한 수정 변경 등은 이하의 특허 청구의 범위에 속하는 것으로 보아야 할 것이다.

(57) 청구의 범위

청구항 1.

정보통신 네트워크 상에서의 경로 탐색 방법에 있어서,

단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 가장 최단 경로를 탐색하기 위해 상기 네트워크 상에 존재하는 모든 경로에 있는 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색 할 시작점을 지정하여 실제 경로 탐색을 수행하도록 하는 제 2 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 2.

상기 제 1 과정은, 노드 0에서부터 노드 N까지 소스로부터 가장 짧은 라우트를 따라 임의의 노드에 쌓인 총 QoS 값들을 초기화 시키는 제 1 과정과;

임의의 최단 경로를 따라 소스로부터 현재 임의의 노드까지 거쳐온 노드를 정리한 집합을 초기화 시키는 제 2 과정과;

현재의 노드와 다음 연결될 노드의 연결 정보를 제공하는 집합을 초기화 시키는 제 3 과정과;

임의의 노드와 그 옆의 임의의 노드 사이에 연결되는 링크의 QoS 값을 초기화 시키는 제 4 과정과;

상기 제 4 과정에서 초기화 된 QoS값을 이용해 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축하는 제 5 과정과;

하여 경로탐색을 수행하도록 하는 제 6 과정을 구비하여;

상기 각 과정을 노드 0부터 노드N까지 반복 수행하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 3.

제 2항에 있어서,

상기 제 5 과정에서 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축할 경우, 현재의 노드에서 다음에 연결될 노드가 자기 자신의 노드가 아니면서 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 4.

제 1항에 있어서,

상기 제 2 과정은, 현재까지 경로 탐색된 노드까지의 총 QoS값을 계산하는 제 1 과정과;

상기 총 QoS값이 계산되면, 이때의 총 QoS값과 다른 경로를 통해 현재 노드까지 도달하여 기 계산되었던 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기하는 제 2 과정과;

상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 경로 탐색 정보를 수정하기 위해, 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 판단하는 제 3 과정과;

상기 판단결과 자신의 노드가 아니면 이 노드를 연결 정보에 추가 등록시키고, 자신의 노드이면 추가 등록을 생략하는 제 4 과정과;

상기 4 과정 후, 상기 제 1과정에서 계산된 총 QoS값을 현재까지의 경로에 대한 총 라우트 QoS값에 대입하고, 현재까지의 노드를 나타내는 집합에 상기 라우트 경로를 등록하는 제 5 과정과;

상기 각 과정이 완료되면 상기 호출된 노드의 다음 노드에 대한 경로 탐색을 위해, 상기 다음 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후 상기 각 과정을 반복 수행하도록 하는 제 6 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 5.

정보통신 네트워크 상에서의 경로 탐색 방법에 있어서,

단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 가장 최단 경로를 탐색하기 위해 상기 네트워크 상에 존재하는 모든 경로에 있는 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색을 수행한 후, 결과 메시지를 상기 호출 대상으로 전송하는 제 3과정을 구비하여, 경로 탐색 시간을 단축하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 6.

제 5항에 있어서,

상기 제 1 과정은 노드 0에서부터 노드 N까지 소스로부터 가장 짧은 라우트를 따라 임의의 노드에 쌓인 총 QoS 값들을 초기화 시키는 제 1 과정과;

상기 임의의 노드에 대한 정보를 가지는 값과, 이 노드와 연결될 다음 노드를 가리키는 값을 초기화 시키는 제 2 과정과;

상기 임의의 노드와 그 옆의 임의의 노드 사이에 연결되는 링크의 QoS 값을 초기화 시키는 제 3 과정과;

상기 제 3 과정에서 초기화 된 QoS값을 이용해 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축하는 제 4 과정과;

상기 초기화가 완료되면 현재 노드가 소스노드에 해당하는지 여부를 판단하는 제 5 과정과;

상기 판단결과 소스노드이면 경로를 나타내는 집합을 현재노드로 초기화하고, 상기 집합의 QoS 값을 '0'로 초기화시킨 다음 경로 탐색 수단을 호출하는 제 6 과정과;

상기 판단결과 소스노드가 아니면 이 노드에 해당되는 값과 이때의 경로를 나타내는 집합 및 값을 이용하여 경로 탐색 수단을 호출하는 제 7 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 7.

제 6항에 있어서,

상기 제 4 과정에서 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축할 경우, 현재의 노드에서 다음에 연결될 노드가 자기 자신의 노드가 아니면서 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 8.

제 5항에 있어서,

상기 제 2 과정은 현재까지 경로 탐색된 노드까지의 총 QoS값을 계산하는 제 1 과정과;

상기 총 QoS값이 계산되면, 이때의 총 QoS값과 다른 경로를 통해 현재 노드까지 도달하여 기 계산되었던 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기하는 제 2 과정과;

상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 경로 탐색 정보를 수정하기 위해, 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 판단하는 제 3 과정과;

상기 판단결과 자신의 노드가 아니면 이 노드를 연결 정보에 추가 등록시키고, 자신의 노드이면 추가 등록을 생략하는 제 4 과정과;

상기 4 과정 후, 상기 제 1과정에서 계산된 총 QoS값을 현재까지의 경로에 대한 총 라우트 QoS값에 대입하고, 현재까지의 노드를 나타내는 집합에 상기 라우트 경로를 등록하는 제 5 과정과;

상기 각 과정이 완료되면 상기 호출된 노드의 다음 노드에 대한 경로 탐색을 위해, 상기 다음 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후, 이 결과 메시지를 상기 경로탐색 수단을 호출한 곳으로 전송하는 제 6 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 9.

정보통신 네트워크 상에서의 경로 탐색 방법에 있어서,

단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 기준 QoS 값을 만족하는 모든 경로를 탐색하기 위해 상기 네트워크 상에 존재하는 모든 경로에 있는 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점과, 기준 QoS 값을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색을 수행한 후, 결과 메시지를 상기 호출 대상으로 전송하는 제 3 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 10.

제 9항에 있어서,

상기 제 1 과정은 노드 0에서부터 노드 N까지 소스로부터 가장 짧은 라우트를 따라 임의의 노드에 쌓인 총 QoS 값들을 초기화 시키는 제 1 과정과;

상기 임의의 노드에 대한 정보를 가지는 값과, 이 노드와 연결될 다음 노드를 가리키는 값을 초기화 시키는 제 2 과정과;

상기 임의의 노드와 그 옆의 임의의 노드 사이에 연결되는 링크의 QoS 값을 초기화 시키는 제 3 과정과;

상기 제 3 과정에서 초기화 된 QoS값을 이용해 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축하는 제 4 과정과;

상기 초기화가 완료되면 현재 노드가 소스노드에 해당하는지 여부를 판단하는 제 5 과정과;

상기 판단결과 소스노드이면 경로를 나타내는 집합을 현재노드로 초기화하고, 상기 집합의 QoS 값을 '0'로 초기화시킨 다음 경로 탐색 수단을 호출하는 제 6 과정과;

상기 판단결과 소스노드가 아니면 이 노드에 해당되는 값과 기준 QoS 값과, 이때의 경로를 나타내는 집합 및 값을 이용하여 경로 탐색 수단을 호출하는 제 7 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 11.

제 10항에 있어서,

상기 제 4 과정에서 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축할 경우, 현재의 노드에서 다음에 연결될 노드가 자기 자신의 노드가 아니면 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 12.

제 9항에 있어서,

상기 제 2 과정은 현재까지 경로 탐색된 노드까지의 총 QoS값을 계산하는 제 1 과정과;

소스노드에서부터 지금 노드 까지의 축적된 총 QoS값이 상기 기준 QoS 값을 만족하는지 여부를 판단하여, 만족하지 않으면 경로 탐색을 종료하는 제 2 과정과;

상기 기준 QoS 값을 만족하면, 총 QoS 값과 다른 경로를 통해 현재 노드까지 도달하여 기 계산되었던 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기하는 제 3 과정과;

상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 경로 탐색 정보를 수정하기 위해, 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 판단하는 제 4 과정과;

상기 판단결과 자신의 노드가 아니면 이 노드를 연결 정보에 추가 등록시키고, 자신의 노드이면 추가 등록을 생략하는 제 5 과정과;

에 상기 라우트 경도를 등록하는 제 6 과정과;

상기 각 과정이 완료되면 상기 호출된 노드의 다음 노드에 대한 경로 탐색을 위해, 상기 다음 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후 이 값과, 기존 QoS 값을 곱해 메시지로 하여, 상기 경로탐색 수단을 호출한 곳으로 전송하는 제 7 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 13.

정보통신 네트워크 상에서의 경로 탐색 방법에 있어서,

상기 네트워크의 계층이 다중레벨인 네트워크 상에서 단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 가장 최단 경로를 탐색하기 위해 상기 다중레벨 네트워크 상에 존재하는 모든 경로에 있는 계층내 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색 수행 시 한번 경유한 계층의 영역은 재 경유하지 않으며, 경로탐색이 수행되면 결과 메시지를 상기 호출 대상으로 전송하는 제 3과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 14.

제 13항에 있어서,

상기 제 1 과정은 노드 0에서부터 노드 N까지 소스로부터 가장 짧은 라우트를 따라 임의의 계층내 노드에 쌓인 총 QoS 값들을 초기화 시키는 제 1 과정과;

상기 임의의 계층내 노드에 대한 정보를 가지는 값과, 이 계층내 노드와 연결될 다음 노드를 가리키는 값을 초기화 시키는 제 2 과정과;

상기 임의의 계층내 노드와 그 옆의 임의의 노드 사이에 연결되는 링크의 QoS 값을 초기화 시키는 제 3 과정과;

상기 제 3 과정에서 초기화 된 QoS값을 이용해 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축하는 제 4 과정과;

상기 초기화가 완료되면 현재 계층내 노드가 소스노드에 해당하는지 여부를 판단하는 제 5 과정과;

상기 판단결과 소스노드이면 경로를 나타내는 집합을 현재 계층내 노드로 초기화하고, 상기 집합의 QoS 값을 '0'로 초기화시킨 다음 경로 탐색 수단을 호출하는 제 6 과정과;

상기 판단결과 소스노드가 아니면 이 계층내 노드에 해당되는 값과 이때의 경로를 나타내는 집합 및 값을 이용하여 경로 탐색 수단을 호출하는 제 7 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 15.

제 13항에 있어서,

상기 제 4 과정에서 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축할 경우, 현재의 계층내 노드에서 다음에 연결될 계층내 노드가 자기 자신의 노드가 아니면서 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 16.

제 13항에 있어서,

상기 제 2 과정은 현재까지 경로 탐색된 임의의 계층내 노드까지의 총 QoS값을 계산하는 제 1 과정과;

상기 총 QoS값이 계산되면, 이때의 총 QoS값과 다른 경로를 통해 현재 노드까지 도달하여 기 계산되었던 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기하는 제 2 과정과;

상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 경로 탐색 정보를 수정하기 위해, 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 판단하는 제 3 과정과;

상기 판단결과 자신의 노드가 아니면 이 계층내 노드를 연결 정보에 추가 등록시키고, 자신의 노드이면 추가 등록을 생략하는 제 4 과정과;

상기 4 과정 후, 상기 제 1과정에서 계산된 총 QoS값을 현재까지의 경로에 대한 총 라우트 QoS값에 대입하고, 현재까지의 계층내 노드를 나타내는 집합에 상기 라우트 경로를 등록하는 제 5 과정과;

상기 각 과정이 완료되면 상기 호출된 계층내 노드의 다음 노드에 대한 경로 탐색을 위해, 상기 다음 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후, 이 결과 메시지를 상기 경로탐색 수단을 호출한 곳으로 전송하는 제 6 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 17.

정보통신 네트워크 상에서의 경로 탐색 방법에 있어서,

상기 네트워크의 계층이 다중레벨인 네트워크 상에서 단일 소스노드에서 단일 목적지 노드로 발생할 수 있는 다수의 경로 중 기준 QoS 값을 만족하는 모든 경로를 탐색하기 위해 상기 다중레벨 네트워크 상에 존재하는 모든 경로에 있는 계층내 노드들의 연결정보를 나타내는 집합 및 서비스 품질(QoS)값을 초기화 시키는 제 1 과정과;

상기 초기화가 완료되면 경로 탐색을 분산처리 하기 위해, 경로 탐색 할 시작점과, 기준 QoS 값을 지정하여 경로 탐색 수단을 호출하고, 상기 경로 탐색 수단에서의 결과 메시지를 수신 대기하는 제 2 과정과;

상기 호출에 따라 경로 탐색을 수행한 후, 결과 메시지를 상기 호출 대상으로 전송하는 제 3 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 18.
제 17항에 있어서,

상기 제 1 과정은 노드 0에서부터 노드 N까지 소스로부터 가장 짧은 라우트를 따라 임의의 계층내 노드에 쌓인 총 QoS 값들을 초기화 시키는 제 1 과정과;

상기 임의의 계층내 노드에 대한 정보를 가지는 값과, 이 계층내 노드와 연결될 다음 노드를 가리키는 값을 초기화 시키는 제 2 과정과;

상기 임의의 계층내 노드와 그 옆의 임의의 노드 사이에 연결되는 링크의 QoS 값을 초기화 시키는 제 3 과정과;

상기 제 3 과정에서 초기화 된 QoS값을 이용해 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축하는 제 4 과정과;

상기 초기화가 완료되면 현재 계층내 노드가 소스노드에 해당하는지 여부를 판단하는 제 5 과정과;

상기 판단결과 소스노드이면 경로를 나타내는 집합을 현재 계층내 노드로 초기화하고, 상기 집합의 QoS 값을 '0'로 초기화시킨 다음 경로 탐색 수단을 호출하는 제 6 과정과;

상기 판단결과 소스노드가 아니면 이 계층내 노드에 해당되는 값과 기준 QoS 값과, 이때의 경로를 나타내는 집합 및 값을 이용하여 경로 탐색 수단을 호출하는 제 7 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 19.
제 17항에 있어서,

상기 제 4 과정에서 자신의 노드 다음에 연결될 노드의 정보를 나타내는 집합을 구축할 경우, 현재의 계층내 노드에서 다음에 연결될 계층내 노드가 자기 자신의 노드가 아니면서 패스가 연결된 노드이면 다음 연결 노드를 표시하는 집합에 추가로 등록하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

청구항 20.
제 17항에 있어서,

상기 제 2 과정은 현재까지 경로 탐색된 계층내 노드까지의 총 QoS값을 계산하는 제 1 과정과;

소스노드에서부터 현재 계층내 노드 까지의 축적된 총 QoS값이 상기 기준 QoS 값을 만족하는지 여부를 판단하여, 만족하지 않으면 경로 탐색을 종료하는 제 2 과정과;

상기 기준 QoS 값을 만족하면, 총 QoS 값과 다른 경로를 통해 현재 계층의 노드까지 도달하여 기 계산되었던 총 QoS값과 비교하여 기 계산되었던 값이 더 작으면 현재 탐색 중이던 경로 탐색을 중단하고 다른 경로 탐색 호출이 있기를 대기하는 제 3 과정과;

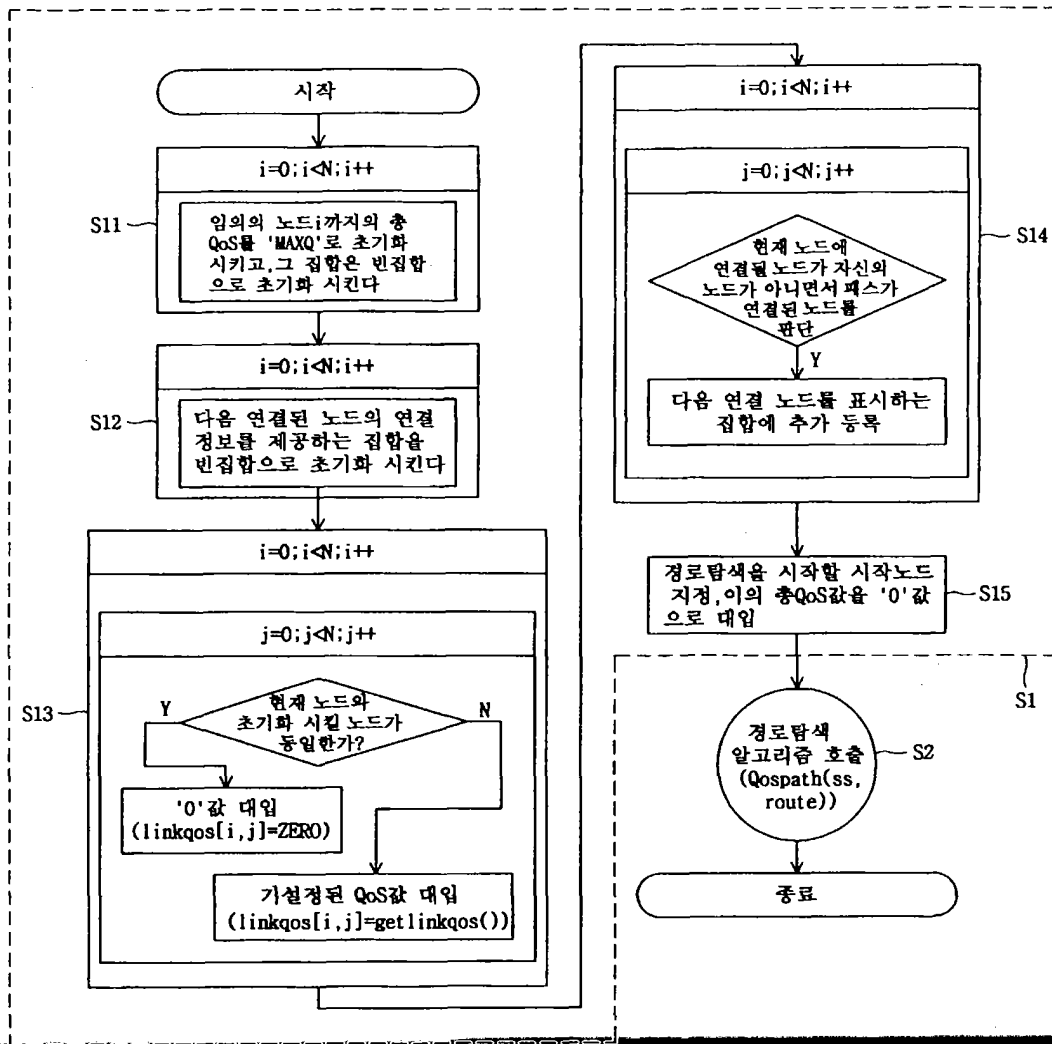
상기 비교결과 현재 도착한 경로의 총 QoS값이 더 작으면 경로 탐색 정보를 수정하기 위해, 현재 경로로 도착한 경로의 마지막 노드가 자신의 노드와 동일한가의 여부를 판단하는 제 4 과정과;

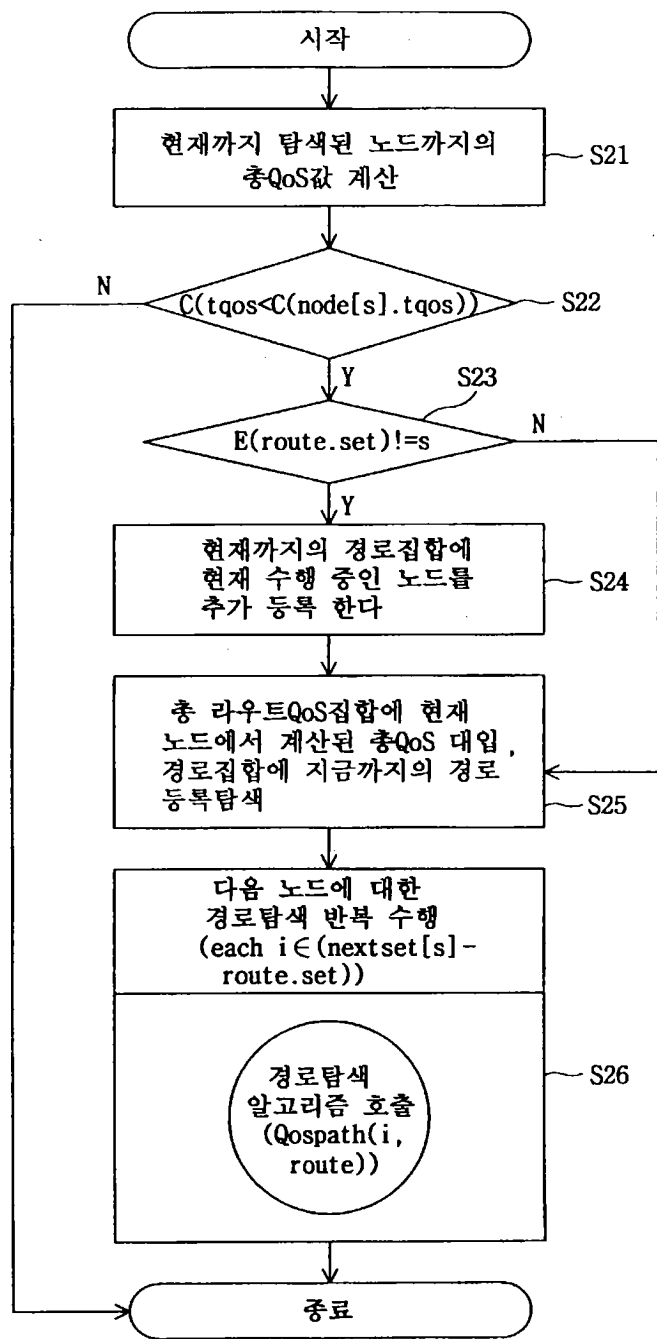
상기 판단결과 자신의 노드가 아니면 이 계층내 노드를 연결 정보에 추가 등록시키고, 자신의 노드이면 추가 등록을 생략하는 제 5 과정과;

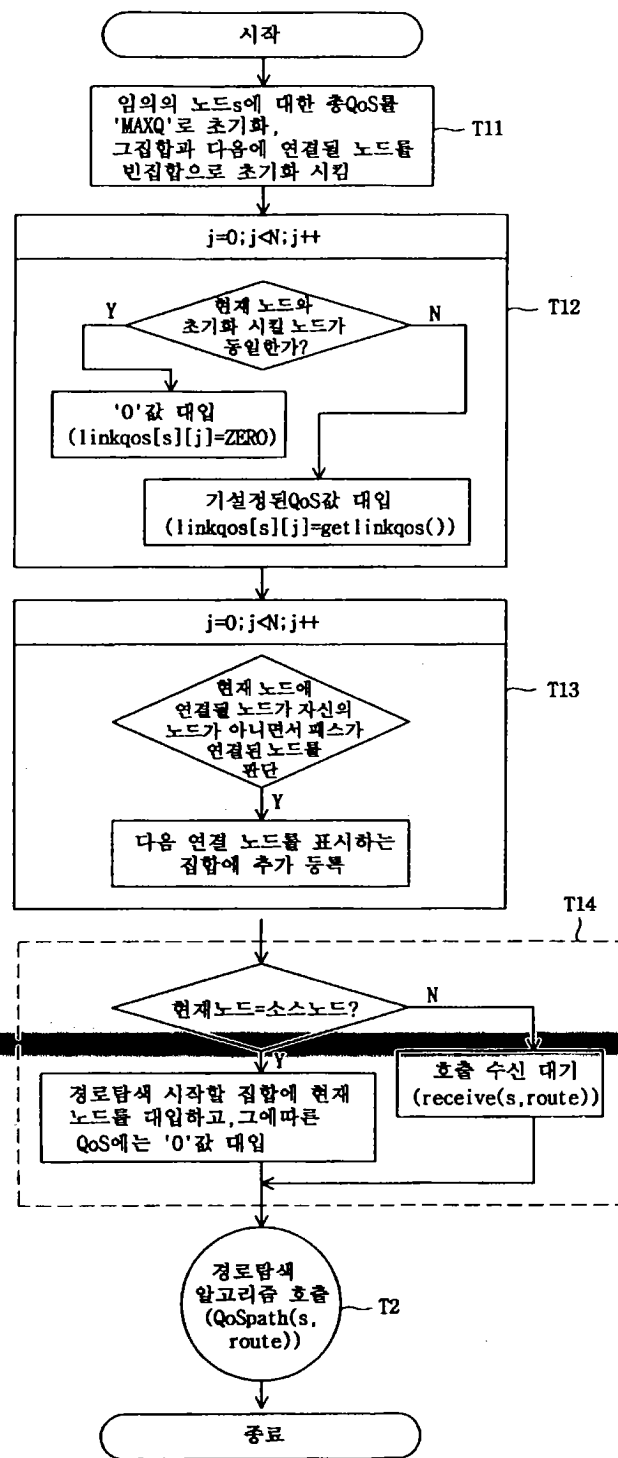
상기 4 과정 후, 상기 제 1과정에서 계산된 총 QoS값을 현재까지의 경로에 대한 총 라우트 QoS값에 대입하고, 현재까지의 계층내 노드를 나타내는 집합에 상기 라우트 경로를 등록하는 제 6 과정과;

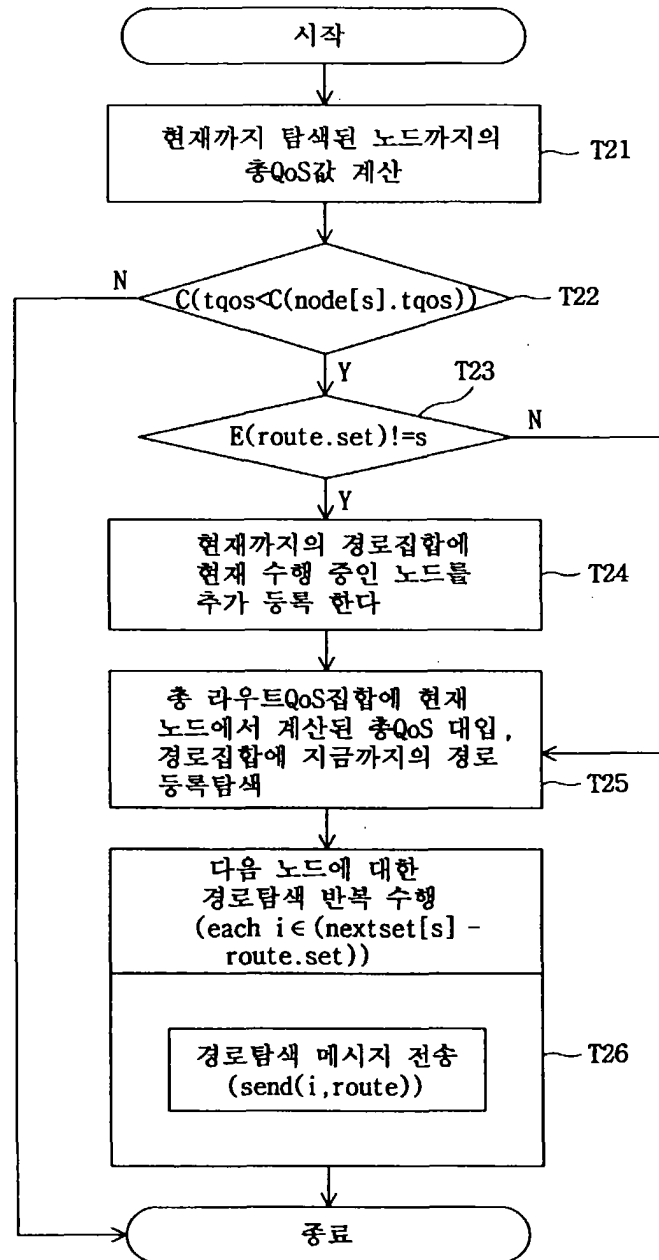
상기 각 과정이 완료되면 상기 호출된 계층내 노드의 다음 노드에 대한 경로 탐색을 위해, 상기 다음 노드에 대한 연결을 나타내는 집합에서 상기 수행되었던 라우트 집합을 뺀 후 이 값과, 기준 QoS 값을 결과 메시지로 하여, 상기 경로탐색 수단을 호출한 곳으로 전송하는 제 7 과정을 구비하는 것을 특징으로 하는 서비스 품질(QoS)을 지원하는 경로 탐색 방법.

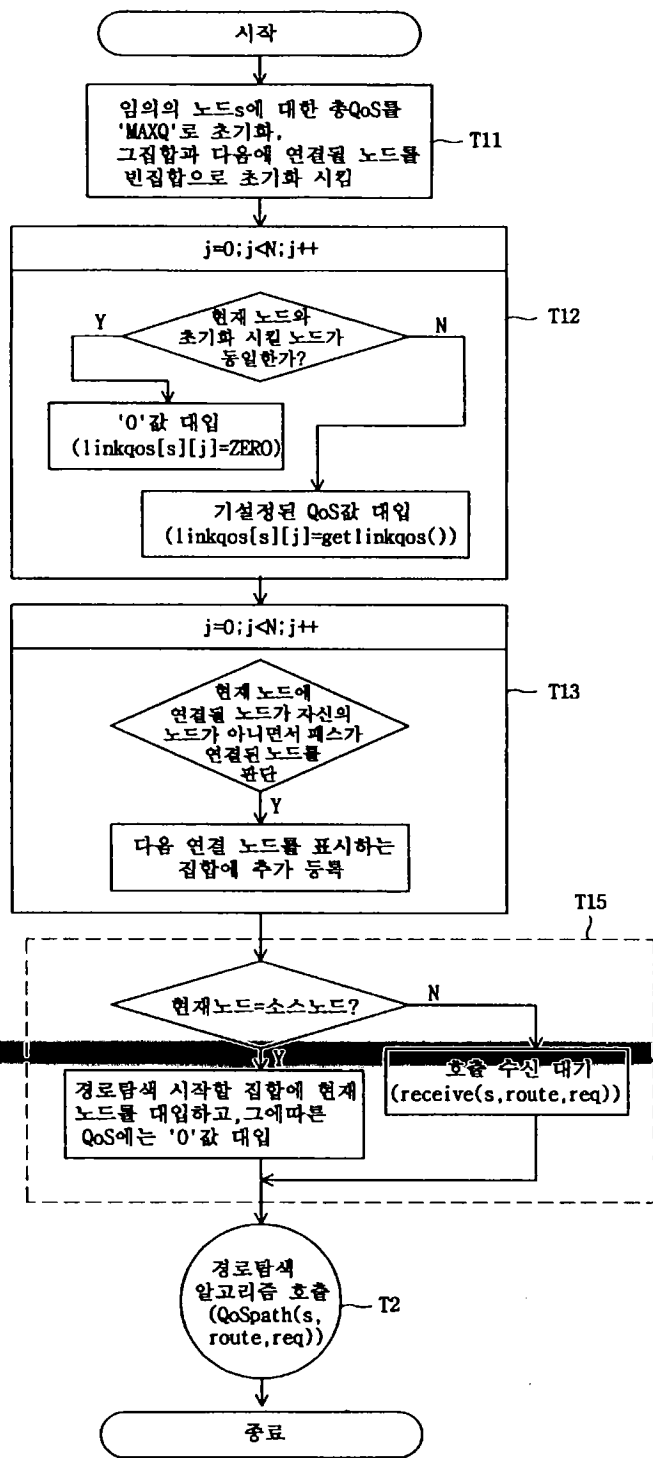
도면

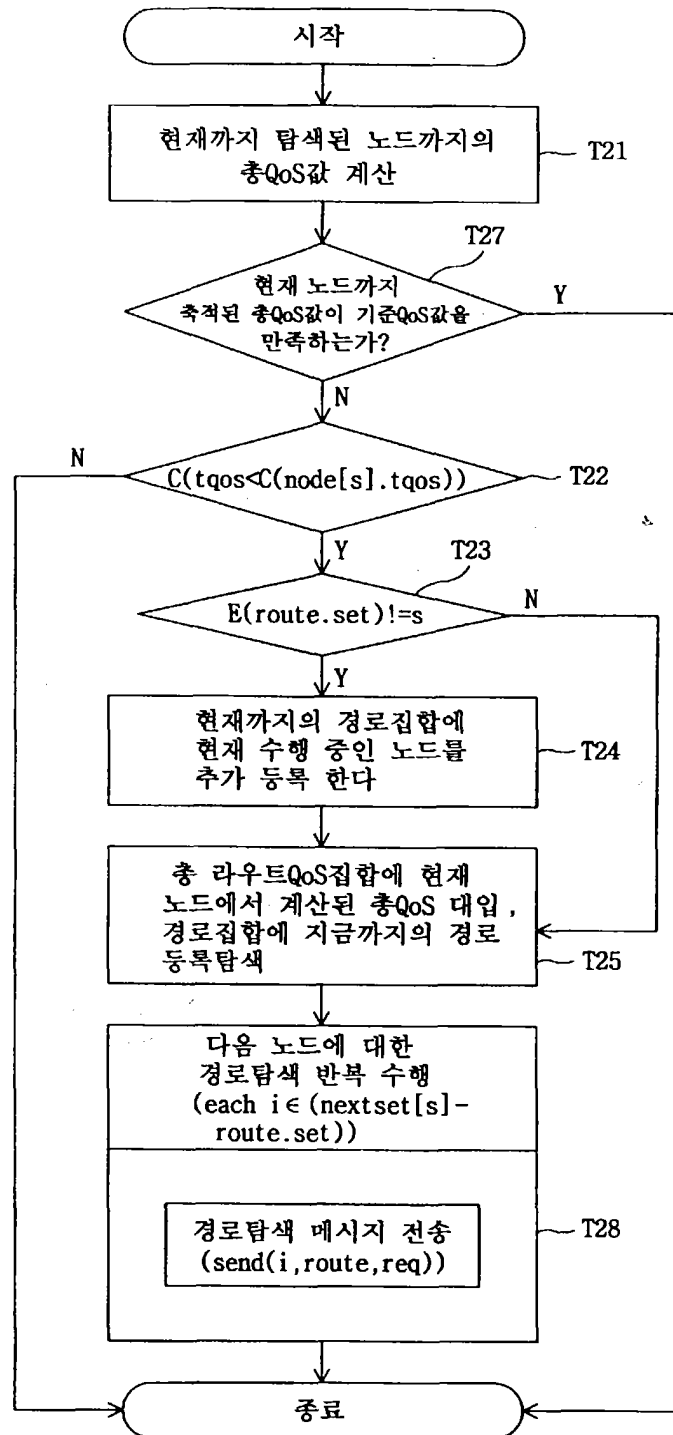


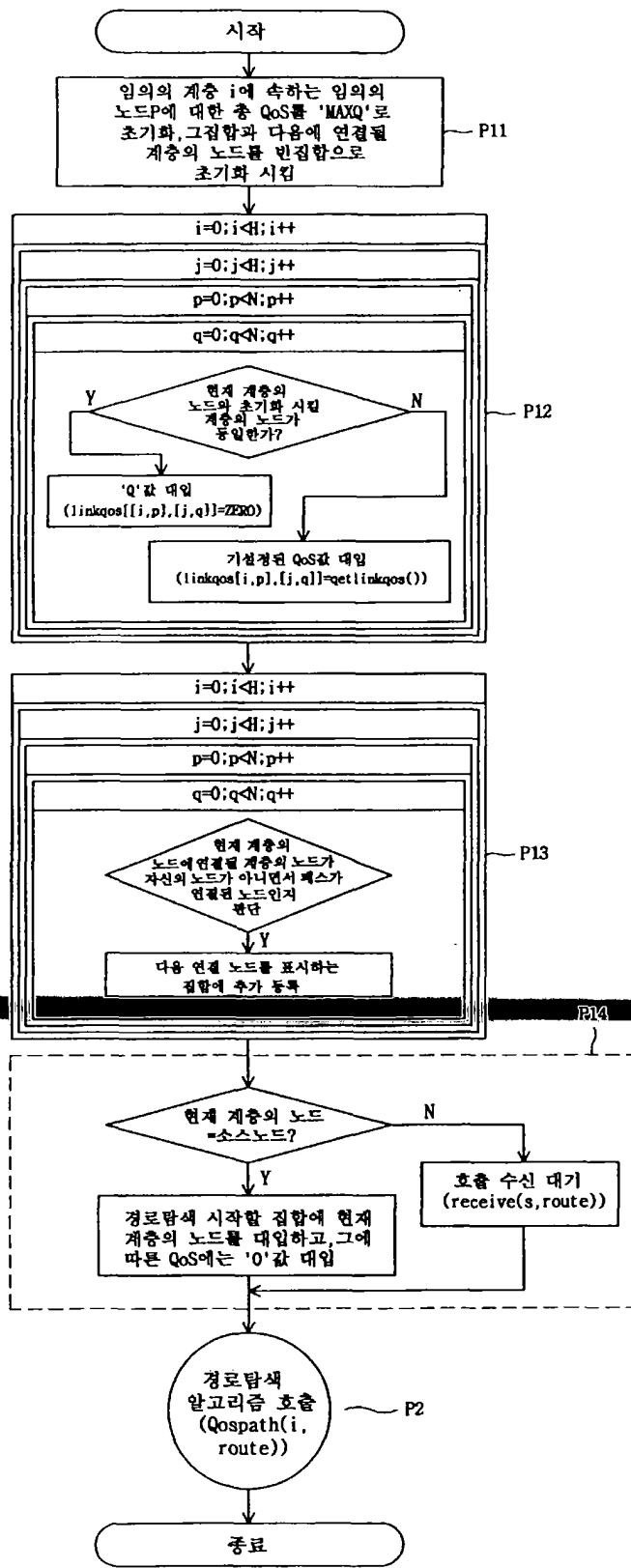


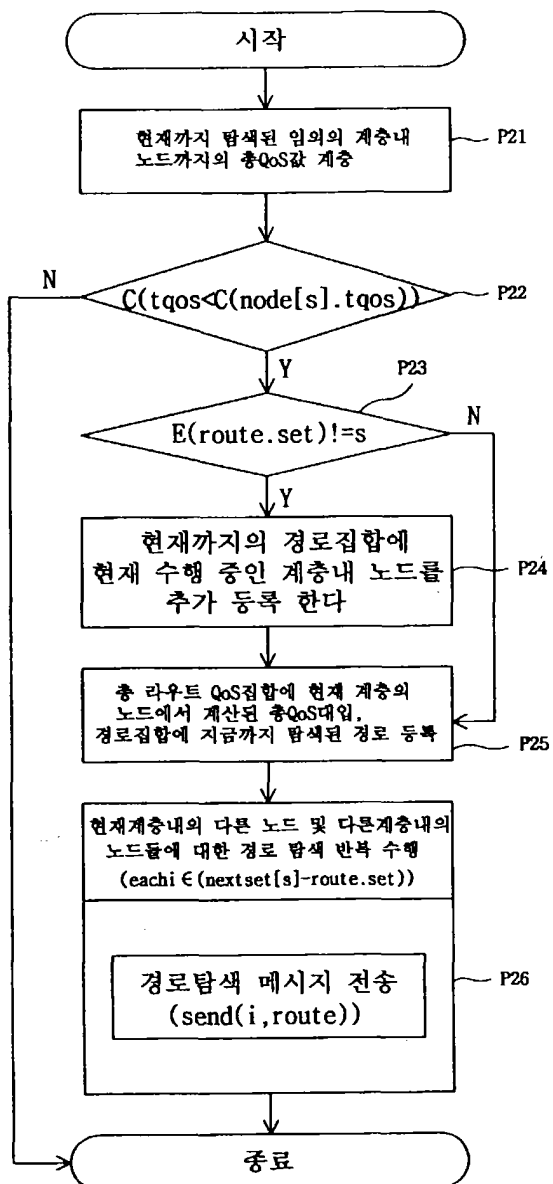


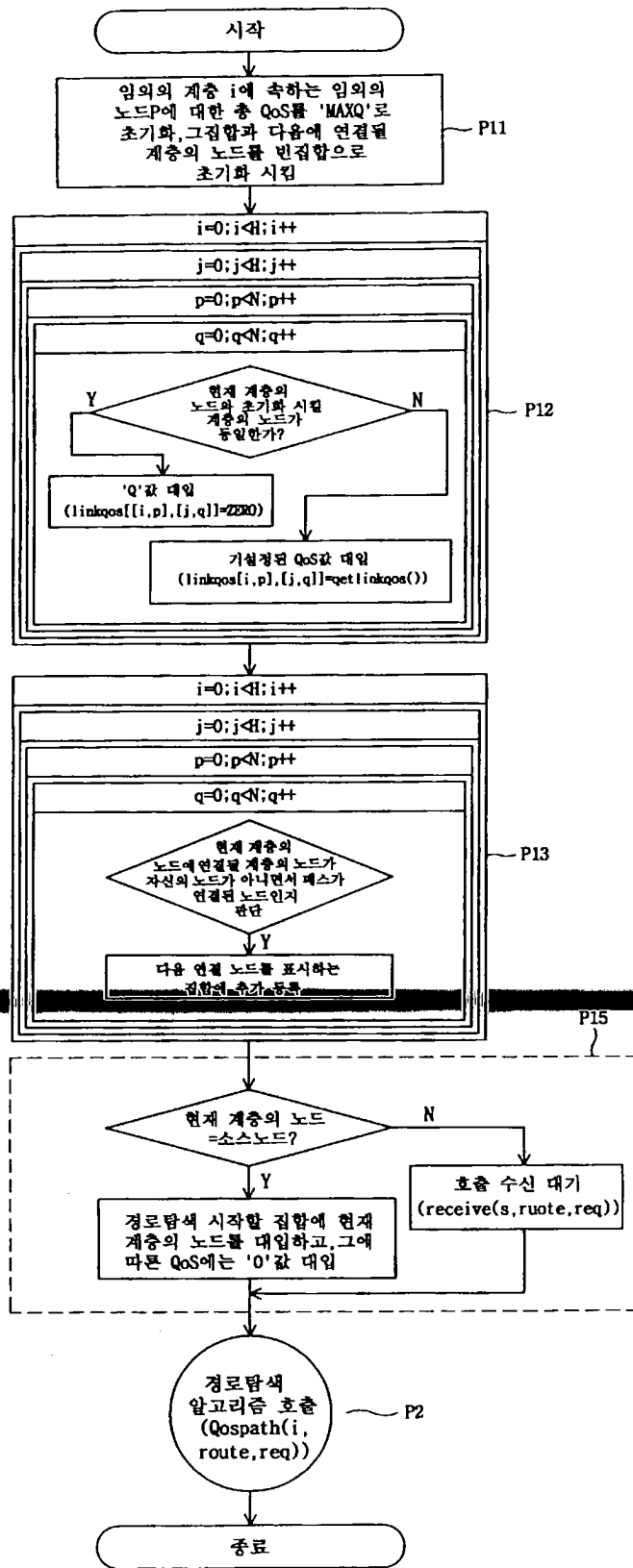












도면 10

